Homework 8 - API

CS 1301 - Intro to Computing - Fall 2021

Important

- Due Date: **Tuesday, March 29th, 11:59 PM**.
- This is an individual assignment. High-level collaboration is encouraged, **but your submission must be uniquely yours.**
- Resources:
  - TA Helpdesk
  - Email TA's or use class Piazza
  - How to Think Like a Computer Scientist
  - CS 1301 YouTube Channel
  - API Handout (in Canvas Files)
  - Installing Pip and Requests (in Canvas Files)
- Comment out or delete all function calls. Only import statements, global variables, and comments are okay to be outside of your functions.
- **Read the entire document before starting this assignment.**

The goal of this homework is for you to enhance your understanding of APIs. The homework will consist of 5 functions for you to implement. You have been given the HW08.py skeleton file to fill out. However, below you will find more detailed information to complete your assignment. Read it thoroughly before you begin.

Hidden Test Cases: In an effort to encourage debugging and writing robust code, we will be including hidden test cases on Gradescope for some functions. You will not be able to see the input or output to these cases. Below is an example output from a failed hidden test case:

```
Test failed: False is not true
```

Written by Arvin Poddar (apoddar32@gatech.edu) & Aryan Verma (aryanverma@gatech.edu)

Helpful Information

For this assignment, we will be using the REST countries API (https://restcountries.com/#api-end-points-v2). **Please read through this documentation, as it will be extremely helpful for completing this assignment.**

For all of your requests, make sure you to use version 2 of the REST countries API (V2), not version 3 (V3.1).

If you make a request with the URL: https://restcountries.com/v2/alpha/usa, you will receive the following response:

```
{
  "name": "United States of America",
  "topLevelDomain": [".us"],
  "alpha2Code": "US",
  "alpha3Code": "USA",
  "callingCodes": ["1"],
  "capital": "Washington, D.C.",
  ...
}
```

If you make a request with an invalid URL, you will receive the following response:

```
{
  "status": 400,
  "message": "Bad Request"
}
```

Please read the description of each problem carefully to ensure you're correctly handling errors.

Highest Population

Function Name: highestPopulation()
Parameters: regionalBloc (`str`)
Returns: country with highest population (`str`)
Description: For a geography class, you are interested in finding the highest population of countries in different regional blocs. Given the name of a regional bloc (`str`), write a function that returns the name of the country with the highest population in that regional bloc (`str`).

Note: You can assume that the regional bloc will always be valid.

```
>>> highestPopulation('saarc')
'India'
```

```
>>> highestPopulation('eu')
'Germany'
```

Common Time Zones

Function Name: commonTimeZones()
Parameters: code1(`str`) , code2(`str`)
Returns: list of common time zones (`list`)
Description: You and your friend live in different countries, but you figure there's a chance that both of you might be in the same time zone. Thus, you want to find out the list of possible time zones that you and your friend could both be in. Given two country codes, write a function that returns a list of the time zones the two countries have in common. Be sure not to include any duplicate time zones. If the two country codes do not have any common time zones, return the string `'No Common Time Zones'` instead.

Note: You can assume that the codes will always be valid.

```
>>> commonTimeZones('can', 'usa')
['UTC-08:00', 'UTC-07:00', 'UTC-06:00', 'UTC-05:00', 'UTC-04:00']
```

```
>>> commonTimeZones('rus', 'chn')
['UTC+08:00']
```

Register Domains

Function Name: registerDomains()
Parameters: companyName (`str`), countryList (`list`)
Returns: list of domain names (`list`)
Description: You've just started a new company and you want to register your company's website in countries around the world. To create a domain name for your website in a country, you'll need to obtain the "top level domain" for that country and add it to the end of your company's name, all in lowercase. Given a list of country names, write a function that returns a list of new domain names for these countries. The order of these domain names should correspond to the original order of `countryList`. If a country name is invalid, simply skip over that country.

```
>>> companyName = 'Google'
>>> countryList = ['france', 'japan', 'canada']
>>> registerDomains(companyName, countryList)
['google.fr', 'google.jp', 'google.ca']
```

```
>>> companyName = 'Amazon'
>>> countryList = ['germany', 'narnia', 'mexico', 'hogwarts']
>>> registerDomains(companyName, countryList)
['amazon.de', 'amazon.mx']
```

Find Country

Function Name: findCountry()
Parameters: capitalList (`list`)
Returns: Dictionary mapping each capital to its country(`dict`)
Description: Given a list of countries' capital cities, write a function that returns a dictionary that maps a capital to its country's name.

Note: Assume that all capital cities provided are valid.

```
>>> findCountry(['tokyo', 'delhi', 'stockholm'])
{'tokyo': 'Japan', 'delhi': 'India', 'stockholm': 'Sweden'}
```

```
>>> findCountry(['paris', 'canberra', 'copenhagen'])
{'paris': 'France', 'canberra': 'Australia', 'copenhagen': 'Denmark'}
```

Most Common Language

Function Name: commonLanguages()
Parameters: regionalBloc (`str`)
Returns: most common language (`str`) or languages (`list`)
Description: As you travel around the world, you want to learn languages that are popular in many countries. Given a regional bloc (`str`), write a function that returns the name of the most common language in that regional bloc. If there is a tie for which language is most common, return a list of all the tied languages **alphabetically** sorted.

Hint: You can find the most common language of a regional bloc by finding the language that is spoken in the most countries out of the countries in that regional bloc.

Note: You may assume that the given regional bloc will always be valid.

```
>>> commonLanguages('cais')
'Spanish'
```

```
>>> commonLanguages('eu')
['English', 'French', 'German']
```

Grading Rubric

Function	Points
highestPopulation()	20
commonTimeZones()	20
registerDomains()	20
findCountry()	20
commonLanguages()	20
Total	**100**

Provided

The `HW08.py` skeleton file has been provided to you. This is the file you will edit and implement. All instructions for what the functions should do are in this skeleton and this document.

Submission Process

For this homework, we will be using Gradescope for submissions and automatic grading. When you submit your `HW08.py` file to the appropriate assignment on Gradescope, the autograder will run automatically. The grade you see on Gradescope will be the grade you get, unless your grading TA sees signs of you trying to defeat the system in your code. You can re-submit this assignment an unlimited number of times until the deadline; just click the “Resubmit” button at the lower right-hand corner of Gradescope. You do not need to submit your `HW08.py` on Canvas.